UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/066,475	02/01/2002	Edward Colles Nevill	1103179-0009	5943

7470 7590 03/24/2010

WHITE & CASE LLP
PATENT DEPARTMENT
1155 AVENUE OF THE AMERICAS
NEW YORK, NY 10036

EXAMINER	
COULTER, KENNETH R	
ART UNIT	**PAPER NUMBER**
2454	

MAIL DATE	DELIVERY MODE
03/24/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

RECORD OF ORAL HEARING

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte EDWARD COLLES NEVILL

Appeal 2009-014323
Application 10/066,475
Technology Center 2400

Oral Hearing Held: March 3, 2010

Before ROBERT E. NAPPI, JOHN A. JEFFERY, and
ST. JOHN COURTENAY III, *Administrative Patent Judges.*

APPEARANCES:

ON BEHALF OF THE APPELLANT:

KEVIN X. McGANN, ESQUIRE
White & Case, LLP
1155 Avenue of the Americas
New York, New York 10036-2787

The above-entitled matter came on for hearing on Wednesday, March 3, 2010, commencing at 9:00 a.m., at the U.S. Patent and Trademark Office, 600 Dulany Street, Alexandria, Virginia, before Paula Lowery, Notary Public.

THE CLERK: Good morning. Calendar Number 1, Mr. McGann.

JUDGE NAPPI: Good morning.

MR. MCGANN: First of all, thank you for taking the time to hear me this morning. The reason we've asked for oral argument on this matter is it appears to me, looking at the record, that there's a fundamental misunderstanding of the Larsen reference and how it works, and how it's being applied to the Applicant's claims.

The Applicant's claims are directed to an apparatus and method for microprocessors that would handle multiple instruction sets. Instruction sets are like binary codes that are used and decoded by the processor to tell the rest of the processor what to do.

So when you have more than one instruction set, you can imagine that you have essentially two languages or two sets of codes; and the processor needs to know what language or what code it's dealing with -- what instruction set. If it doesn't know the code, it would be like a cryptographer that you give a coded message but you don't tell them the code. They need to know the code to decode it with. So that's why the processor needs to know what instruction set it's dealing with.

JUDGE COURTENAY: So you use a high-order bit to designate one of these instruction sets?

MR. MCGANN: One example in the Applicant's disclosure is to use a high-order bit that is reserved that does not use an address.

The main difference to focus on the Larsen versus the Applicant's invention is that Larsen relies upon, and we can see this -- I might direct you to Figure 2 in Larsen, for example, which I've blown up on the second set of materials I handed you.

Larsen relies upon this idea of partitioning memory so that the high-order addresses, or the high-order section of I-Store that's there in Figure 2, which is where instructions are stored, has one type of instructions, Type 2 instructions.

The other portion of memory has a different type of instructions, Type 1 instructions, okay?

So the gist of what Larsen is talking about is I've partitioned memory. I've separated this out so that I know when I draw instructions from Sector 1 memory that they're going to be instruction Type 2. When I draw them from anywhere else in memory, they'll be instruction Type 1.

To do that, Larsen provides part of the address -- at least part of the address -- of the instruction. It has a location in memory. It has an address just like our houses have an address. If I want to know where somebody lives or where an instruction resides, I give the address.

JUDGE COURTENAY: So the upper three-most bits are used to designate the eight partitions in Larsen?

MR. MCGANN: It could be partitioned in any number of ways.

JUDGE COURTENAY: Well, they have binary zero to seven here, so it's eight partitions.

MR. MCGANN: Correct.

JUDGE COURTENAY: We understand that.

MR. MCGANN: Okay. If I need to go back to any -- I wasn't familiar with

the Court's understanding of technology, so I figured I'd provide a little background. If I jump ahead and you'd like explanation, please stop me.

Again, let's look at Figure 2 and look at how this is described in Larsen. The portion that the Examiner cites is in Column 5, line 52 to Column 6, line 16, that I point out in that first slide.

The section describes how instructions are decoded, and it makes it clear that the identity of the instruction set is necessary to properly decode the instruction set.

In order to identify the type of instruction, I need to use its address.

So necessarily in Larsen there are bits from the instruction address that are used to identify the instruction sets to specify how to decode the instructions.

We can see that in the section the Examiner has cited against the Applicant's claims where it says the contents of the IDSR-10, which includes part of the address, are taken together with the instruction to provide input to the instruction decoding logic, which is the IDM-5.

So the decoding is always done using part of the address and the instruction itself. Without knowing where the instruction came from in Larsen, you can't decode it properly.

If you look at Figure 2, for example, and I tried to blow up a particular portion of Larsen Figure 2, and this is -- it has the blue arrows on it.

If we look, Larsen explains that the instruction address resides in the IAR register. So that's Item 3 in Figure 2. That instruction address is provided to whatever logic, it's not really described, but logic that would access the instruction storage area on the instruction address box, that's the IAP.

So on the left side of Figure 2, we see the instruction address register sends

the address to the instruction address plus. That's the whole address -- all of it. That goes up to the I-Store 2 to tell you where the instruction resides. Also, at least a portion of that is reused in the IDSR, that's 10 on Figure 2. So you have this -- and this is what we tried to summarize in the first slide shown in our Brief.

JUDGE NAPPI: Stop for a moment. Is that portion of the address still an address?

MR. MCGANN: Which portion?

JUDGE NAPPI: You said it takes at least a portion of the address to --

MR. MCGANN: It is.

JUDGE NAPPI: So a portion of an address is still an address?

MR. MCGANN: I'm sorry, no, it's not. I misunderstood your question.

JUDGE COURTENAY: Why isn't it?

MR. MCGANN: Because it wouldn't specify the unique location. Let me give you an example.

JUDGE COURTENAY: Okay.

MR. MCGANN: My house address is 43. What I've effectively done here is by taking the high-order bits, it would be telling you my address is 4 something.

JUDGE COURTENAY: Or it could be your address is 43 Robins Lane, Vienna, Virginia, and if I just give you 43 Robins Lane and you happen to already know you're in Vienna, Virginia, that would be a sufficient address to get me to your house, wouldn't it?

Your complete address would include your house, the street, and the state. A short address or portion of the address might just be your house number and your street. Why doesn't that same kind of rationale apply to

Larsen?

MR. MCGANN: Because when I'm dealing with an instruction, I need to know one unique location to pull the exact correct instruction. It may be a portion of the address, as Your Honor mentioned, but it's just that. It can only be a portion.

An example in Larsen Figure 2 -- if I have 111, that tells me I'm somewhere in the upper eighth of the instructions.

JUDGE COURTENAY: Yes, and then the second part of that address tells you where to look in that I-Store, right?

MR. MCGANN: Right, but it would be used together to uniquely identify one spot in the instructions.

JUDGE JEFFERY: I think the Examiner is taking the position that the last five bits, if you will, of this eight-bit address in the I-Store corresponds to the subset -- I'm particularly referring to Claim 65, by the way.

MR. MCGANN: Okay.

JUDGE JEFFERY: It's probably best to jump to the broadest claim first. The subset that's being recited here, as I understand the Examiner's position, the Examiner is taking the position that the last five bits correspond to that. As I understand your position in the Reply Brief on page 8, you indicate those bits only indicate part of an address, not the entire address, and there's eight possible addresses, in effect, that could be identified here by the top three bits.

So you have the same last five bits and three different top three bits, you're going to get eight different addresses identified is what I understand. So your position is that because there are eight possible combinations of

addresses in taking the Examiner's position, you couldn't possibly decode the correct address, is my understanding.

MR. MCGANN: I think Your Honor has said and as I said, you would not uniquely identify an address in this example where you have it divided in eighths. You'd have one of eight possibilities.

JUDGE JEFFERY: One of eight possibilities and when you go to the claim, the claim simply says "an ordered set of bits" -- Claim 65 -- "wherein a subset of the ordered set of bits identifies . . . ," and I'm going to emphasize this word, "*an* address of *an* instruction." Does that preclude an *incorrect* address of an *incorrect* instruction?

MR. MCGANN: I wouldn't think that somebody would read it that way to include an incorrect address because then it wouldn't function.

If I continually access the wrong instructions, as we pointed out in the Reply Brief, just picking one out of eight, you'd be wrong 90 percent of the time. You'd be wrong 90 percent of the time by design, not to mention if there were any errors in actual operation.

I would assume that somebody skilled in the art would not read a claim in a way that would fail 90 percent of the time.

JUDGE JEFFERY: So you have a one in eight chance of getting it right, basically, taking this Examiner's position with the last five bits being the subset.

I would also add that Claim 65 doesn't have any functionality, does it? I mean, we're not doing anything with these bits, unlike some of the other claims where we're using the information to decode or the like. Here, we're just identifying things, are we not?

So my point being: the fact that the Examiner's position as to the last five bits -- certainly in the Examiner's mind, these things are identifying -- these last five bits are identifying some type of instruction, albeit partial.

MR. MCGANN: I guess my response would be that they don't identify an instruction.

JUDGE JEFFERY: An address --

MR. MCGANN: They would identify a range of possible instructions.

JUDGE JEFFERY: One of eight.

MR. MCGANN: Which makes them not be an address. That subset isn't an address because it doesn't tell you where to go to get that instruction.

JUDGE COURTENAY: Actually, in the reference they're referred to specifically as high-order addresses at Column 5, Line 47. It says these are identified as having the high-order address beginning with "111" in binary. So they're specifically referred to in the reference as high-order addresses -- the upper most three-fifths -- that identify the eight partitions.

MR. MCGANN: That, in my view, exactly supports the Applicant's argument. Those bits are required as part of the address.

JUDGE COURTENAY: So it's by virtue of identifying the different partitions -- the eight partitions in Figure 2 -- it illustrates that the upper-most partition, Partition 7 in binary "111," is reserved for Type 2 instructions. The rest of the partitions are Type 1 instructions.

It also identifies the type of instruction set.

MR. MCGANN: That's exactly the problem I have with applying Larsen to the Applicant's claims. I think in one or more ways each of these claims requires that you can't use the address, or part of the address, to specify the instruction set or instruction type; and Larsen can't do that.

JUDGE COURTENAY: Can you point to your language in Claim 65 that precludes that?

MR. MCGANN: Sure. In Claim 65 we have -- it refers to what would be in the program counter. You have the ordered set of bits, okay? There's a subset of that ordered set that identifies the address of the instruction, okay? There's at least one bit in that set that identifies the instruction set. The last part of that is the important part where at least one bit is not a member of the subset.

If we look at Larsen, you have these eight bits that are all the address. You have three of those eight are repeated and used again to identify the instruction set. So there is

no -- there can't be at least one bit that identifies the instruction set, meaning those three, that are both part of the address and identify the instructions.

JUDGE JEFFERY: The Examiner is simply taking the position that the top three bits plus the five -- all eight of the bits, the so-called "ordered set of bits," and that the subset in the claim is the last five bits.

So the top three clearly indicate the type of instruction that you have in terms of the partition that you're in in the I-Store, whether it's a "l l l" or what have you, but the last five bits in isolation, in the Examiner's mind, seem to indicate an address, albeit partial.

MR. MCGANN: I guess my response would be that is, as you said, a partial address. It's not an address. I think the claim says that subset has to identify an address.

JUDGE NAPPI: Counsel, it seems to me we're arguing over what the definition of an address is. You seem to be saying an address is complete,

all the information you need; and we're saying any information that gets you to a location.

What I'd like to know, since we're looking at the claims now, where is your definition of address in your specification to find that it's got to be in such a manner that precludes what the Examiner has said?

MR. MCGANN: Well, one example --

JUDGE NAPPI: A definition.

MR. MCGANN: A definition?

JUDGE NAPPI: Yeah, I think we're arguing back and forth over examples. I'd like to see a definition that nails it.

MR. MCGANN: One example might be in Column 4, beginning around Line 5 -- actually, it's line 6. It refers to the program counter register.

JUDGE NAPPI: Um-hum.

MR. MCGANN: The program counter register addresses memory.

JUDGE NAPPI: Okay.

MR. MCGANN: So the address that it would use is the information that would be residing in the program counter.

If you look at Column 4 along side of Figure 1 of the Applicant's, you see it's Item 130, the PC? That's the portion of that that other than the part that's labeled T, the portion of that is what would contain the address.

JUDGE NAPPI: Okay. You have just given us this nice figure of Larsen, which has your portion of T-1 and PC together on the right-hand side, and then Larsen that has the high-order bits and lower bits.

Again, we keep coming back to the same issues. We have examples here, and the example in Larsen -- it seems to me the Examiner is saying that the low-order address is an address for a certain portion of memory.

You're saying, well, that's not the complete address, and I'm trying to understand why we have to interpret your claim as being to the complete address.

MR. MCGANN: That section in Column 4 refers to the program -- I'm looking at approximately Line 8, where it talks about program counter controller increments the value in the program counter. As each instruction is executed, a new instruction must be fetched.

JUDGE NAPPI: Okay.

MR. MCGANN: So the address is to fetch an instruction.

JUDGE NAPPI: Okay.

MR. MCGANN: It's not to point to a general 1/8th, for example, of the addressable memory.

JUDGE NAPPI: But is it fetching the instruction?

MR. MCGANN: I'm sorry?

JUDGE NAPPI: In Larsen, is it fetching the instruction in that 1/8th of the memory?

MR. MCGANN: Not without using those three high-order bits. It wouldn't fetch the instruction.

JUDGE NAPPI: Okay.

MR. MCGANN: It couldn't fetch an instruction without using those three high-order bits, which is why it never -- the whole point of Larsen is it relies upon partition in the memory so that address tells you -- and this is even in the section that the Examiner cites to us.

I'm reading from Column 5, and this is -- the section the Examiner cites is Line 52 to Column 6, Line 16 of Larsen. I'm reading particularly at Column 6, Lines 11-14.

It says: "The decoding of the instruction depends not only the contents of the IDR-1, which is the instruction itself, but on the region of the I-Store from which the instruction was fetched."

So that region, which is specified by part of the address, is used twice. It's used to give you the address -- 111 followed by five or six digits that specifies the address.

JUDGE NAPPI: But doesn't that section also say that the low-order bit is used to fetch an instruction from IDR-1?

MR. MCGANN: I'm sorry, where are you referring to?

JUDGE NAPPI: Once you're in IDR-1, doesn't the low-order bit identify an instruction to be fetched from IDR-1?

MR. MCGANN: IDR-1 in Larsen actually contains the instruction itself. I-Store contains the --

JUDGE JEFFERY: Yeah, there's two things. The address is stored in the I-Store, and the instruction is stored in the IDR. The actual instruction itself is stored in the IDR.

MR. MCGANN: Correct.

JUDGE JEFFERY: Right. So the question is, fetching an instruction: Is that dependent on the address of the instruction?

MR. MCGANN: It would be because you would have -- think of the memory that you use. This could be many megabits or more of memory. You need to tell the controller where to go in that memory to get the instruction you want to execute.

There might be -- the example of pipelining, where you have multiple instructions being fetched at one time, but it would still be a specific set or address -- a specific range in the I-Store -- that you want to address.

JUDGE JEFFERY: Counsel, I wanted to ask you -- our time is limited here. I do want to ask you about one thing that caught my eye in Column 6 of Larsen, you know, beginning around Line 3, where it talks about the content of the IDSR-10 and the IDR-1 are taken together to provide a look-up address within the IDM-5. Fine.

This next couple of sentences though caught my eye. "Only specific portions of the undecoded instruction in IDR may be needed according to the format of the instruction being fetched. The other portion of the input address to the IDM is the output from the IDSR-10, which identifies the fetch-from address in the I-Store."

What does that mean? Only specific portions of the undecoded instruction in the IDR may be needed? Are we talking -- does that bolster the Examiner's position some how some way in terms of just needing the last five bits, if you will? It's not the full number of bits?

MR. MCGANN: With all due respect, that's the instruction, not its address.

JUDGE JEFFERY: Okay.

MR. MCGANN: The Examiner's argument is that I can use some part of an address, and it's an address. I think the response is it's not an address in the instruction.

JUDGE JEFFERY: Okay.

MR. MCGANN: When you have an instruction, you have -- separately, they're all ones and zeros, but you have parts of the instruction that tell the processor what to do. Other parts may tell is what the operands are. So the instruction is: and the operands are 3 and 4.

JUDGE JEFFERY: Why do you think Larsen said the other portion of the input address right after that sentence? I know the preceding sentence says

". . . and portions of the instruction," so that would be the instruction itself, but then he goes on to say ". . . the other portions of the input address." Is that just an anomaly in the wording?

MR. MCGANN: It's a specific way the implementation of Larsen -- instead of having --

JUDGE JEFFERY: Why did he call it "the other portions"? That's my question.

MR. MCGANN: The way Larsen is set up with the IDM-5 is -- you have two ways to decode an instruction, a least two ways. One is hard wire the logic, ands and operands where I apply electrical signals to it and I get a definite output. Another is I have a look-up table where the bits of the instruction -- not the address, the bits of the instruction -- that tell you it's an add instruction. Tell me go to this place in the instruction decoder to determine what to do. That's completely separate from where the instruction is stored in I-Store and memory. It's a function of the specific implementation.

JUDGE JEFFERY: So the terminology "the other portion of the input address" to the IDM-5, "the other portion of the input address" would be the top three bits? Is that what we're talking about?

MR. MCGANN: No.

JUDGE JEFFERY: The "other portion" --

MR. MCGANN: It's not an address that's used to address the instruction at all. It's a look-up table --

JUDGE JEFFERY: Well, it says the output of the IDSR-10. Isn't that the top three bits?

MR. MCGANN: Oh, I'm sorry. The top three bits of the IDSR --

JUDGE JEFFERY: Right, let's make sure we're on the same sheet of music here. I'm starting on Line 8 of Column 6, and I'll quote:

"The other portion of the input address to the IDM-5 is the output from the IDSR-10 which identifies the fetched-from address."

That would be the top three bits, would it not?

MR. MCGANN: Yes.

JUDGE JEFFERY: Okay. The only reason I bring this up is that terminology "other portion of the input address" comes right after where he talks about specific portions of the undecoded instruction. Putting those two things together, it sounds like one portion could be the last five bits, and the other portion, the top three bits.

What I think I hear you saying is, they're apples and oranges, in effect. You've got addresses, you have instructions, portions of each aren't the same.

MR. MCGANN: You do, and if I could beg your indulgence for two minutes to try to explain what Larsen Figure 2 shows, maybe I could clear that up.

JUDGE JEFFERY: Well, I understand how this works. I don't know if that particular passage is reflected in the functionality here specifically.

You know, we're taking address information along with the instruction itself and using that as a look up in the instruction decode memory to decode the instructions. So I think we all understand how this works.

MR. MCGANN: Right.

JUDGE JEFFERY: It was sort of a peculiar terminology, I thought, in the reference that it was talking about “the other portions” kind of thing.

MR. MCGANN: Well, I think that may also suggest -- although it's in a different context because it's in decoding, the address really refers to what would be a unique location, rather than a region.

JUDGE JEFFERY: I have one other question for you, if I may. There seems to be, from the record reading this over, a bit of inconsistency in how the Examiner is interpreting -- I'm going to Claim 1 now, the last clause of Claim 1 -- the memory access controller.

MR. MCGANN: Yes.

JUDGE JEFFERY: Frankly, the Examiner's rejection on page 9 of the answer is a little tough to sort out exactly how the Examiner is mapping the elements to the memory address controller. Specifically in Item (e) on page 9 of the Answer, he refers to Figure 2 and then cites three different elements: 3, 10 and 2, which are the IAR, IDSR, and the I-Store, respectively, there. In your Brief on page 18 of your Brief, you tend to refer to the memory access controller, line 9 on page 18 of the Brief, you say: "Memory Access Controller, the address-resolving logic 6, in Larsen."

So do we have an inconsistency there in terms of how we're interpreting that element?

MR. MCGANN: I believe my reference to the address was on the logic 6, and that on page 18 you referred to was incorrect.

It is, in fact, not really shown in the Larsen figure. It's whatever logic would use the address in the I-Store, not decoding logic.

JUDGE JEFFERY: All right.

MR. MCGANN: I apologize, that was my mistake. I apologize for the confusion.

JUDGE JEFFERY: Okay.

MR. MCGANN: I appreciate you pointing out "(e)" because I think it's not just an inconsistency. I think the Examiner has misunderstood that whatever memory access controller there is, the access is the I-Store.

The Examiner in responding to (e), Claim 1, says those bits in the top three bits, the access controller would not be responsive to that; and that's clearly not the case in Larsen. As we see in Figure 2 and the description, all the bits of the address are put on the address box.

For example, if we look at Figure 2, we have the instruction addresses in the IAR-3. Those bits -- all of them -- go up to the IAD to the I-Store.

So those bits are not the controller that controls how to decode the instruction -- I'm sorry, how to retrieve the instruction from memory. It can't be not responsive to those bits. It's definitely responsive. Those three bits are included in the address. So Larsen couldn't disclose that.

JUDGE COURTENAY: But isn't it true once you have the partition selection by the high-order three bits, you have eight partitions of instructions? The upper-most partition is the Type-2 instructions, and the rest are Type 1.

Once you have a given partition selected, the low-order address bits determine an instruction in a given partition, isn't that true? Would you agree to that?

The low-order address bits uniquely address an instruction within a given partition that has been selected by the high-order bits.

MR. MCGANN: But it can only be done in combination with those high-order bits. I think even as Your Honor just phrased it, once those are identified by the high-order bits, the high-order and low-order bits are used together to determine the address.

JUDGE COURTENAY: Well, it seems your position is the subset of the ordered set of bits, being the lower order address bits, doesn't uniquely identify an address.

MR. MCGANN: Okay.

JUDGE COURTENAY: But it does within the context of a given partition. The low-order address bits uniquely identify an address, an instruction, once the partition has been selected by the high-order address bits.

The high-order address bits have a dual function. They select the partition, but they also, by selecting the different partitions, they identify the type of instruction.

MR. MCGANN: I agree with the latter part of what Your Honor said. The high-order address bits -- because you call them address bits -- they are the high-order address bits, do both functions.

JUDGE COURTENAY: They have dual functions as I see it.

MR. MCGANN: I'm sorry?

JUDGE COURTENAY: They have a dual function.

MR. MCGANN: Correct, and that's exactly what we're saying distinguishes the Applicant's invention. Because if you look, for example, at Figure 2, I believe it is of the Applicant's invention, that may be the best place it's illustrated -- Figures 2 and 3.

You see it refers to the memory address and then separately this bit that tells you what the instructions are. So you have an address, and you have something else that tells you the instruction type.

You don't need to reuse -- this provides a non-trivial advantage in application. You don't have to partition memory because I don't care where the instructions of different types are stored.

It would be important to a software designer. Imagine if I had partitioned 1/8th of the memory, and the instructions that I wanted to use of that particular type spilled over?

JUDGE COURTENAY: I understand the advantage you're discussing, but we're interested in what the broadest reasonable construction of your claim reads on the reference. That's our concern.

MR. MCGANN: I think the broadest reasonable construction couldn't read on it because an address of an instruction is its location in memory. It's a place, not a region.

JUDGE COURTENAY: Don't you agree we have a partial address in these low-order address bits that uniquely identify an instruction in a given partition?

MR. MCGANN: They provide a partial address.

JUDGE COURTENAY: A partial address is an address. If you consider an address the genus and you consider a partial address the species, a species anticipates a genus.

MR. MCGANN: But they wouldn't identify the address of an instruction.

JUDGE COURTENAY: Within a given partition it would. Once that partition is selected by the higher order address bits.

MR. MCGANN: But that very point is you have to use those high-order address bits to know the region and combine -- so the address is the combination of those bits.

It has to include the high-order bits, otherwise I could be in any position in the memory.

JUDGE COURTENAY: That depends on how we construe an address. Whether an address is just an address within a given partition is sufficient to

meet your claim, or whether we have to look at all the address bits as uniquely identifying a given location of an instruction within all eight partitions.

MR. MCGANN: Well, I think the fair reading of the claims is it identifies an address and instruction. Taking Your Honor's point as to 65, which is the first one the Examiner dealt with, the language of the claim says the ordered sets of bits identifies an address and instruction, not a regional I-Store, not one of many possible instructions. It's an address.

JUDGE COURTENAY: But again, the low-order bits do identify an address of an instruction within the context of a single function.

MR. MCGANN: Not alone. They can't do it alone because built into your very question, Your Honor –

JUDGE COURTENAY: I agree that the partition has to be selected by the high-order three bits.

MR. MCGANN: So to go back to Judge Nappi's example, if I had a house number and street name in the city, the address of my house is all of those things.

The address of an instruction -- what I understand Your Honor is saying, if I knew, for example, that I was in Alexandria, Virginia, and knew further what neighborhood in Alexandria I was in, I might be able to give a street number and name and somebody would find the location.

But that's not how instructions work. I've already built into that assumption that I know I'm in Alexandria, I know the region I'm in. That's the address. It's what uniquely identifies that building. This building is 600 Dulany Street.

JUDGE COURTENAY: Well, if Dulany Street corresponds to the high-order bits, say we have three bits that uniquely identify Dulany Street to use your analogy, 600 would be the low-order bit that uniquely identifies the location within Dulany Street.

The 600 Dulany Street is an address and is Dulany Street. It's just a broader address when you're looking at the street, or in the case of the reference the three high-order bits identify a unique partition.

MR. MCGANN: Right, I would think of the address in this case and as reflected in the claim language, it's an address of an instruction. The analogy would be it's an address -- it depends on what the address is supposed to be of.

If it's a city, Alexandria might be enough. If it's an instruction, the instruction has a specific location in memory that's defined by the high-order bits and the low-order bits in Larsen.

JUDGE COURTENAY: So I understand your position to be that an address, as recited in Claim 65, can only be a unique address within the context of all eight partitions in the Larsen reference shown on Figure 2. That's your position, where you need the whole address.

MR. MCGANN: I believe that's correct.

JUDGE COURTENAY: I understand your position.

MR. MCGANN: It has the address of an instruction.

JUDGE NAPPI: Any more questions?

JUDGE COURTENAY: No.

JUDGE JEFFERY: No.

JUDGE NAPPI: Thank you very much.

Whereupon, the proceedings at 9:40 a.m. were concluded.